COMPLETE LISTING OF CLAIMS

IN ASCENDING ORDER WITH STATUS INDICATOR

1. (currently amended) A multiplier comprising:

means for receiving at least two operands, A and B;

means for generating a product of said at least two operands, wherein said generating means is arranged to enable both signed and unsigned multiplication; and

wherein said means for generating implements a modified Baugh-Wooley algorithm for signed multiplication that generates a product of operands A and B ~~as the result of~~ according to $((S_A*S_B*2^{2N-2})+(S_A*(-1)*2^{N-1}*B')+(S_B*(-1)*2^{N-1}*A')+(A'*B'))$, wherein N is the number of bits in each operand, S_A is the sign bit for operand A, S_B is the sign bit for operand B, A' is the bits of operand A excluding its sign bit, and B' is the bits of operand B excluding its sign bit.

2. (currently amended) The multiplier of claim 1 wherein said means for generating is further configured to perform unsigned multiplication by generating a product of said operands A and B ~~as a result of:~~ according to:

$((S_A*S_B*2^{2N-2})+(S_A*2^{N-1}*B')+(S_B*2^{N-1}*A')+(A'*B'))$.

3. (original) The multiplier of claim 1 wherein said modified Baugh-Wooley algorithm translates a signed operand to an unsigned operand.

4. (original) The multiplier of claim 1 wherein said generating means is implemented having a static design.

5. (original) The multiplier of claim 1 wherein said multiplier is operable at a frequency of 1 GHz or greater.

6. (original) The multiplier of claim 1 wherein said multiplier is operable to perform multiplication when multiplication is enabled for said multiplier, and wherein said multiplier is operable to perform population count for a received operand when population count is enabled for said multiplier.

7. (original) The multiplier of claim 1 further comprising:

means for identifying whether signed or unsigned multiplication is desired.

8. (currently amended) A system comprising:

multiplier for generating a product of at least two operands;

said multiplier comprising a linear summation array for summing partial products of said at least two operands, wherein said linear summation array is arranged to enable both signed and unsigned multiplication;

wherein said linear summation array is implemented to generate a product of operands A and B ~~as a result of~~ according to $((S_A*S_B*2^{2N-2})+(S_A*(-1)*2^{N-1}*B')+(S_B*(-1)*2^{N-1}*A')+(A'*B'))$ when performing signed multiplication, wherein N is the number of bits in each operand, S_A is the sign bit for operand A, S_B is the sign bit for operand B, A' is the bits of operand A excluding its sign bit, and B' is the bits of operand B excluding its sign bit; and

wherein said linear summation array is implemented to generate a product of operands A and B ~~as a result of~~ according to $((S_A*S_B*2^{2N-2})+(S_A*2^{N-1}*B')+(S_B*2^{N-1}*A')+(A'*B'))$ when performing unsigned multiplication, wherein N is the number of bits in each operand, S_A is the most significant bit for operand A, S_B is the most significant bit for operand B, A' is the bits of operand A excluding its most significant bit, and B' is the bits of operand B excluding its most significant bit.

9. (original) The system of claim 8 further comprising at least one processor.

10. (original) The system of claim 8 wherein said linear summation array is implemented as an even-and-odd structure having a static design.

11. (original) The system of claim 8 wherein the resulting columns of said linear summation array correspond to the input pitch of the operands input to the multiplier.

12. (original) The system of claim 11 wherein two operands having 16 bits each are input to the multiplier, said linear summation array resulting for said two operands having size 16 by 14.

13. (currently amended) A method of performing multiplication comprising the steps of:

receiving at least two operands, A and B, in a multiplier;

identifying whether signed or unsigned multiplication is desired;

if signed multiplication is desired, then computing the product of said at least two operands A and B ~~as a result of~~ according to $((S_A*S_B*2^{2N-2})+(S_A*(-1)*2^{N-1}*B')+(S_B*(-1)*2^{N-1}*A')+(A'*B'))$, wherein N is the number of bits in each operand, S_A is the sign bit for operand A, S_B is the sign bit for operand B, A' is the bits of operand A excluding its sign bit, and B' is the bits of operand B excluding its sign bit; and

if unsigned multiplication is desired, then computing the product of said at least two operands A and B ~~as a result of~~ according to $((S_A*S_B*2^{2N-2})+(S_A*2^{N-1}*B')+(S_B*2^{N-1}*A')+(A'*B'))$, wherein N is the number of bits in each operand, S_A is the most significant bit for operand A, S_B is the most significant bit for operand B, A' is the bits of operand A excluding its most significant bit, and B' is the bits of operand B excluding its most significant bit.

14. (original) The method of claim 13 further comprising:

using a common set of computational resources for computing the product of operands A and B for both signed and unsigned multiplication.

15. (original) The method of claim 13 further comprising:

identifying whether multiplication or population count is enabled; and

using said multiplier to perform population count for a received operand when population count is enabled.